ISSN: 1692-7257 - Volumen 1 - Número 19 - Año 2012

Revista Colombiana de Tecnologías de Avanzada

Recibido: 02 de febrero de 2011
Aceptado: 21 de marzo de 2011

STAGE: A GRADUATE INFORMATION MANAGEMENT SEMANTIC AGENT

STAGE: UN AGENTE SEMÁNTICO PARA LA GESTIÓN DE INFORMACIÓN DE EGRESADOS

PhD. Jaime A. Guzman, Ing. Durley Torres
Ing. Arlex D. Martínez

Universidad Nacional de Colombia. Sede Medellín, Facultad de Minas
Escuela de Sistemas. Medellín, Antioquia, Colombia.
Tel.: 57-7-5685303, Fax: 57-7-5685303, Ext. 144
E-mail: {jaguzman, idtorresp, admartinezg}@unal.edu.co

Abstract: This article describes the design process and implementation of a semantic agent (called STAGE: *Semantic TAsk aGEnt*). This agent can automatically manage graduate information used in the accreditation process of the Universidad Nacional de Colombia, Medellín. This article describes details of the specification process of ontology knowledge representation associated with the information of graduates, a brief description of query formulation and inference mechanisms of agent that allow it to reason about the information handled.

Keywords: Semantic agent, Ontology, semantic information management, accreditation process.

Resumen: Este artículo describe el proceso de diseño e implementación de un agente semántico (denominado STAGE: *Semantic TAsk aGEnt*). Este agente permite gestionar de manera automática la información de egresados utilizada en el proceso de acreditación de la Universidad Nacional de Colombia, sede Medellín. En este artículo se describen detalles del proceso de especificación de la ontología de representación del conocimiento asociado con la información de los egresados, una breve descripción de la formulación de consultas y los mecanismos de inferencia del agente que le permiten razonar sobre la información manejada.

Palabras clave: Agente semántico, Ontología, gestión de información semántica, proceso de acreditación

1. INTRODUCCIÓN

Este trabajo recopila los resultados de investigación asociados al desarrollo de un agente semántico, denominado STAGE por sus siglas en inglés *Semantic TAsk aGEnt*, el cual fue llevado a cabo en el marco del proceso de acreditación de la Universidad Nacional de Colombia, sede Medellín (UNAL); de manera específica, el agente automatiza el proceso de gestión de la información asociado con los egresados, de dicha universidad.

El agente semántico que se propone específicamente tiene asignadas las tareas de: (i) interpretación semántica de la información recuperada dentro del contexto de una ontología de dominio OWL, la cual especifica el conocimiento concerniente a la información asociada a cada egresado; (ii) almacenamiento del conocimiento semántico obtenido dentro de una estructura persistente construida a partir de JENA y MySQL. Finalmente, (iii) consulta e inferencia, a través de SPARQL y Pellet sobre la base de conocimiento

que a partir de la ontología instanciada con la información de cada egresado almacenado.

La solución desarrollada alrededor del agente semántico implementado reúne las tecnologías de la web semántica y los sistemas multiagente, campos de investigación de notable interés en la actualidad, estas son materializadas a través de un modelo prototipo constituido fundamentalmente por el agente semántico y un conjunto de agentes de prueba desarrollados de manera paralela con el fin de experimentar los procesos asociados a la negociación e intercambio de mensajes entre agentes.

El presente documento se organiza como sigue. En la sección 2 se describe la ontología (OWL) de conocimiento desarrollada, la sección 3 presenta el diseño y arquitectura del agente semántico, en la sección 4 se expone el proceso de implementación y pruebas del prototipo. Y finalmente en la sección 5 se presentan las conclusiones y trabajos futuros.

2. ONTOLOGÍA DE EGRESADOS

Como parte del trabajo realizado, fue diseñada la ontología 'OGIR' (por las siglas en ingles *Ontology Graduate Information Retrieval*), con el fin de conceptualizar la información asociada a los egresados, los conceptos y relaciones establecidas entre conceptos. Dicha ontología, está completamente ajustada, a la información recopilada dentro de los tradicionales formatos de Excel, que actualmente gestiona la universidad de manera manual.

Las clases definidas, hacen referencia, principalmente a conceptos propios de la producción investigativa del egresado. En esta línea existe, por ejemplo, las clases *award*, *project*, *publication*… Otro conjunto de clases hacen referencia a la información personal y académica del egresado, para ello, se definieron clases como: *Personal_Data*, *Curriculum_Program*, entre otros; adicionalmente son importadas clases (y propiedades) propias de otros entornos semánticos de amplia aceptación en la comunidad académica e investigativa, al punto de ser estándares de representación semántica. Tal es el caso del lenguaje *FOAF*, que constituye una recopilación del vocabulario asociado al ámbito de información personal de un individuo. De este estándar, se importaron clases como *Person* y *Company*.
La Fig. 1. presenta un aparte de la ontología diseñada e implementada.

Con las propiedades, a manera general, se pretende asociar el egresado, con sus respectiva producción académica, investigativa y datos personales, para ello se construyeron, relaciones del tipo: *has_Performance*, *has_personalData*, *isAssociatedCurriculumProgram* respectivamente. Otro estandar del que se importaron algunas relaciones corresponde al *DublinCore*, el cual se ocupa de asignar las características propias a cada clase tales como dc:title, dc:date, dc:tittle, etc.).

```
<owl:Class rdf:ID="StateWork">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-s
    <rdfs:label xml:lang="en">StateWork</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="CurriculumProgram">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-s
    <rdfs:label xml:lang="en">CurriculumProgram</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="PersonalData">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-s
    <rdfs:label xml:lang="en">PersonalData</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Company">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-s
    <rdfs:label xml:lang="en">Company</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="HeadQuarters">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-s
    <rdfs:label xml:lang="en">HeadQuarters</rdfs:label>
    <owl:Restriction>
        <owl:onProperty rdf:about="#isAssociatedheadquarters" />
        <owl:someValuesFrom rdf:about="#CurriculumProgram" />
    </owl:Restriction>
</owl:Class>
<owl:Class rdf:ID="Faculty">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-s
    <rdfs:label xml:lang="en">Faculty</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Performance">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-s
    <rdfs:label xml:lang="en">Performance</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Award">
    <rdfs:subClassOf rdf:resource="#Performance"/>
    <rdfs:label xml:lang="en">Award</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Project">
    <rdfs:subClassOf rdf:resource="#Performance"/>
    <rdfs:label xml:lang="en">Project</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Publication">
    <rdfs:subClassOf rdf:resource="#Performance"/>
    <rdfs:label xml:lang="en">Publication</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="EditorialCommittee">
    <owl:equivalentClass rdf:resource="http://xmlns.com/foaf/0.1/
    <rdfs:subClassOf rdf:resource="#Performance"/>
```

Fig. 1. Definición de clases OGIR

Como valor agregado al conocimiento explicito representado por las estructuras expuestas, se añadieron un conjunto de propiedades (*ObjectProperty*), que permiten generar inferencias adicionales sobre la información explícitamente registrada; para ello, se usaron elementos propios del OWL tales como: *owl:equivalentClass*, *owl:inverseOf*, *owl:tansitiveProperty*, entre otras. Con ellas, se hace posible tras la aplicación de un proceso de razonamiento, inferir nuevas relaciones, que no han sido explícitamente declaradas, pero que pueden ser deducidas. Por ejemplo: el concepto *ogir:CurriculumProgram* es asociado a través de la propiedad *owl:equivalentClass* con el concepto *ogir:academic program*.

En la Fig. 2. se muestra la definición de algunas propiedades incorporada al interior de OGIR.

```
<owl:ObjectProperty rdf:ID="isAssociatedheadquarters">
    <owl:inverseOf rdf:resource="#hasParent"/>
    <rdfs:domain rdf:resource="#CurriculumProgram"/>
    <rdfs:range rdf:resource="#Faculty"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Faculty_isAssocited">
    <rdfs:domain rdf:resource="#Faculty"/>
    <rdfs:range rdf:resource="#CurriculumProgram"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="has_Performance">
    <rdfs:domain rdf:resource="#Graduated"/>
    <rdfs:range rdf:resource="#Performance"/>
</owl:ObjectProperty>
```

Fig. 2. Definición de propiedades OGIR

3. DISEÑO DEL AGENTE SEMÁNTICO

Dentro del diseño de STAGE, se ha considerado la construcción de una ontología de comunicación, que permita unificar los conceptos asociados a las tareas de este agente.
Tal especificación facilita la interconexión del agente semántico, con otro u otros agentes.

3.1 Ontología de comunicación

El uso de un vocabulario común (ontología), permite la interpretación coherente e inequívoca de los mensajes intercambiados entre los agentes que comparten tal vocabulario, de este modo, los agentes pueden validar semánticamente un mensaje, entenderlo, localizar y recuperar información específica contenida en él, y en consecuencia realizar de manera eficiente las tareas asociadas a tales mensajes.

En este orden de ideas, como parte del agente semántico implementado, se define a través de la ontología pertinente el dominio específico de acción para tal agente, y los posibles tipos de mensaje que este está en capacidad de interpretar. De manera particular, para el agente semántico, se han definido tres tipos de mensaje: i) consulta, representado por el predicado *Query*, el cual a través de la expresión SPARQL asociada solicita la realización de la consulta dentro de la base de conocimiento del agente semántico. ii) resultado, el cual a través del predicado *ResultSet*, permite enviar la ontología respuesta a una consulta realizada. Y finalmente iii) almacenamiento, el cual a través del predicado *Store*, le indica al agente que se ha generado una solicitud de almacenamiento dentro de la base de conocimiento, asociada a la ontología de egresado. Estos mensajes en consecuencia son relacionados con los comportamientos propios del agente, los cuales son detallados en las secciones siguientes.

3.2 Actividades del agente

El agente semántico implementado, posee un comportamiento reactivo, es decir, permanece en un estado de espera de mensajes externos de solicitud, para realizar sus acciones particulares. Basados en los tipos de mensajes citados en la sección anterior, estas acciones se pueden agrupar en dos grupos de tareas: i) tareas de consulta y respuesta. Y ii) tareas de almacenamiento. El primer grupo de tareas, está orientado a la ejecución de consultas SPARQL y al razonamiento semántico, al interior de la base de conocimiento y el retorno de los resultados para la respectiva consulta. El segundo grupo de tareas tienen como propósito el almacenamiento y actualización de información del egresado dentro de la base de conocimiento. Las subtareas de ambos grupos de actividades se especifican en la Fig. 3.

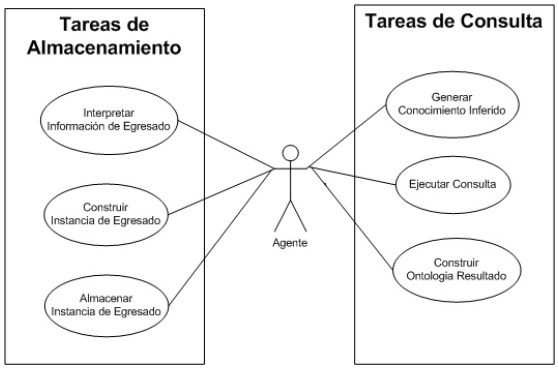

Fig. 3. Casos de uso agente semántico

3.3 Arquitectura del agente semántico

Para llevar a cabo las tareas planteadas para el agente semántico, en el diseño fueron considerados tres módulos básicos los cuales permiten la realización de las tareas de consulta –respuesta y almacenamiento de manera concurrente. Tales módulos son: i) *modulo constructor de ontología*: cuando el agente recibe una solicitud de almacenaje, interpreta la información XML recibida del egresado, para ello utiliza el componente traductor el cual a través de la lectura de las respectivas reglas de traducción permite llevar tal información desde el vocabulario de origen, a la especificación particular de la ontología de egresados, esto le permite al agente ser de propósito general, así, a través de la respectiva configuración el agente puede almacenar información de los egresados a partir de diferentes fuentes (agentes recuperadores, buscadores, entre otros). ii) *modulo gestor de la base de conocimiento*: Una vez construida la instancia de la ontología de egresado, el modulo gestor de base de

conocimiento almacena de manera persistente dicha ontología dentro de una base de datos, indexando tal ontología con el número de identificación del egresado, a través de este índice de es posible recuperar la ontología de un egresado en particular.

iii) módulo motor de consulta: para el caso en el cual el mensaje recibido este orientado a una solicitud de consulta, es recuperada la instancia de ontología correspondiente al identificador del egresado asociado a la consulta realizada, esta instancia previa ejecución de la consulta SPARQL es procesado por el componente de inferencia, el cual permite a través de las reglas y propiedades OWL presentadas en tópicos anteriores generar una ontología extendida con conocimiento adicional inferido. Esta ontología extendida es enviada al componente ejecutor el cual toma la consulta realizada y la aplica sobre tal ontología, finalmente el componente interprete de resultados, toma los resultados obtenidos a partir de la consulta y devuelve un documento RDF donde conformado por los recursos y propiedades inmersos dentro del conjunto de resultados. En la Fig. 4 se presenta la estructura del agente.

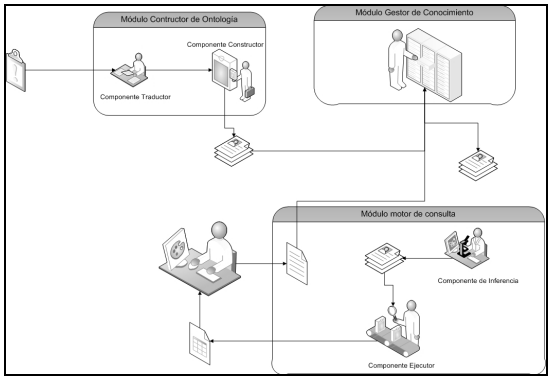

Fig. 4. Arquitectura del agente semántico

4. IMPLEMENTACIÓN DEL PROTOTIPO

STAGE, se ha desarrollado sobre la plataforma JADE (*java agent developer framework*) versión 3.4, tal plataforma está acorde con los estándares de FIPA, y proporciona las estructuras necesarias para la creación y comunicación entre agentes, adicionalmente ofrece herramientas adicionales, para el registro, búsqueda y prueba de los mismos.

4.1 Prueba y validación

Los agentes por definición son elementos colaborativos, en particular para el agente implementado, su naturaleza lo hace altamente dependiente a estímulos externos (mensajes de solicitud consulta/almacenaje), así, para validar el intercambio de comunicación y los comportamientos propios del presente agente semántico, fue necesaria la implementación de dos agentes de prueba, los cuales a través del uso de la ontología de comunicación propia del agente semántico, permiten al usuario generar mensajes de solicitud para consulta y almacenaje.

En consecuencia el primer agente de prueba implementado recibe la información del egresado desde el usuario, construye un documento XML simple con esta información y lo envía al agente STAGE utilizando para ello el códec SL provisto por JADE para la codificación de mensajes desde una ontología de comunicación. En la Fig. 5 es presentada la interfaz de usuario para el primer agente de prueba, en la Fig.6. Se presenta una captura del agente sniffer en el cual se visualiza el mensaje enviado desde el agente de prueba hacia el agente STAGE.

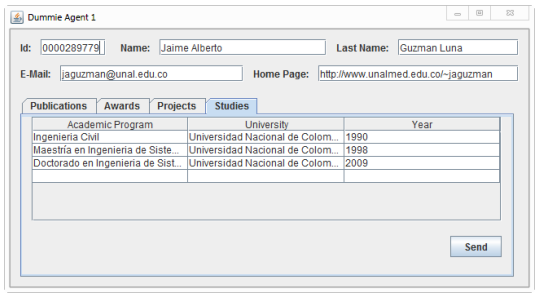

Fig. 5. Interfaz del agente de prueba 1

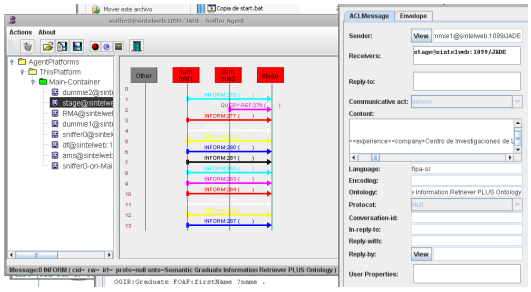

Fig. 6. Intercambio de mensajes agente de prueba - STAGE

Como valor agregado a la construcción de este agente, se implementa de manera paralela la ontología con las reglas de traducción entre el documento XML generado y la especificación de la ontología de egresados usada por el agente STAGE. En la Fig. 7 se muestra un aparte de la ontología de traducción implementada.

```
<translation:TranslationRule rdf:ID="rule01">
    <translation:XMLTag>Graduado</translation:XMLTag>
    <translation:Resource rdf:about="http://www.medellin.unal.edu.co/sintelweb/segir/schema/SEGIRSchema.owl#Graduate"/>
</translation:TranslationRule>
<translation:TranslationRule rdf:ID="rule02">
    <translation:XMLTag>Nombre</translation:XMLTag>
    <translation:Resource rdf:about="http://xmlns.com/foaf/0.1/#firstName"/>
</translation:TranslationRule>
<translation:TranslationRule rdf:ID="rule03">
    <translation:XMLTag>Apellido</translation:XMLTag>
    <translation:Resource rdf:about="http://xmlns.com/foaf/0.1/#lastName"/>
</translation:TranslationRule>
<translation:TranslationRule rdf:ID="rule04">
    <translation:XMLTag>id</translation:XMLTag>
    <translation:Resource rdf:about="http://www.medellin.unal.edu.co/sintelweb/segir/schema/SEGIRSchema.owl#has_identity"/>
</translation:TranslationRule>
```

Fig. 7. Ontología de traducción

El segundo agente de pruebas, está orientada a la consulta sobre la base de conocimiento a través del agente STAGE, en este sentido, el agente de pruebas desarrollado, posee una interfaz de usuario que permite el envió de consultas SPARQL y la presentación de los resultados obtenidos a partir del envío de tal solicitud. En la Fig 8. Se presenta la interfaz de usuario para el agente de prueba 2, con la cual se realiza una consulta SPARQL sobre el egresado identificado con el id '0000289779'.

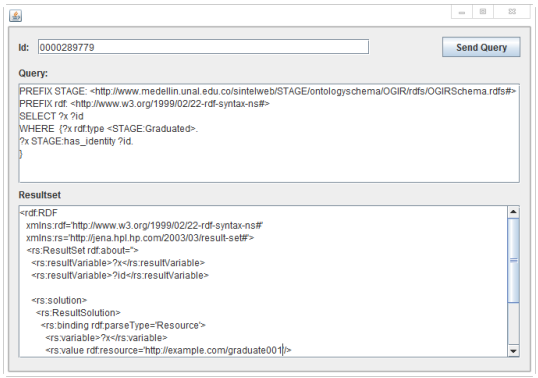

Fig. 8. Interfaz del agente de prueba 2.

4.2 Almacenamiento

La principal tarea del agente semántico implementado como se justifico en tópicos anteriores consiste en la gestión de la información semántica, materializada en documentos OWL, esto se logra con el uso del framework Jena para web semántica de Java, como parte del mecanismo de gestión, este permite el almacenamiento persistente de los documentos llevándolos a una estructura relacional con el asocio de MySQL. De este modo es posible consultar, editar e incluso eliminar documentos OWL relacionados a los egresados almacenados por el agente.

El proceso de almacenamiento es iniciado una vez es recibido un documento XML con la información asociada al egresado, este documento es interpretado en términos de la ontología OGIR, para ello el componente traductor usa las reglas de traducción asociadas a la ontología de traducción ilustrada en la Fig. 7, una vez se construye el documento OWL, este es enviado para almacenarse dentro de la base de datos MySQL, indexando tal documento por el número de identificación del usuario frente a su perfil en Colciencias.

4.3 Inferencia

Como valor agregado a la plataforma de consulta provista por Jena, es acoplado el razonador semántico Pellet, con el cual es posible obtener resultados más completos sobre información implícita inferida a partir de la definición de propiedades y clases de OWL, descrita en la sección 2 (*owl:equivalentClass*, *owl:inverseOf*, *owl:tansitiveProperty*).

El mecanismo del razonador, consiste en realizar la consulta sobre un documento extendido, en el cual, las relaciones implícitas descritas en la definición de la ontología OGIR son instanciadas de manera explícita dentro tal documento de manera temporal, a continuación la consulta SPARQL es realizada sobre tal documento, adquiriendo la totalidad de los resultados (información explicita e implícita).

5. CONCLUSIONES

El rápido crecimiento de la información de la Web y la heterogeneidad de la misma, en la actualidad convergen en un nuevo paradigma para la representación del conocimiento, la Web semántica, como solución para la búsqueda y recuperación de información, se ha implementado el agente STAGE, un agente semántico, que mediante el uso de las tecnologías asociadas a la Web semántica, permite la gestión del conocimiento asociado al dominio específico de los graduados de una institución educativa determinada, el caso particular de implementación está representada por la UNAL, sede Medellín.

6. TRABAJO FUTURO

Como trabajo futuro se propone incorporar el agente STAGE dentro de un sistema multiagente enmarcado en las tareas de búsqueda, almacenaje, consulta y presentación de la información de egresados obteniendo la información desde la Web. El sistema planteado será evaluado en diferentes dominios de conocimiento.

REFERENCIAS

Berners-Lee T., Hendler J. y Lassila O. (2001), *The Semantic Web*. Scientific American, No. 284 **Vol. 5**, pp. 34–43.

Fernández J., Polo A., Arévalo L. y Cerrillo E. (2005), *Recuperación del conocimiento basada en contexto: Una aplicación en la arqueología (Arqueonto)*, JISBD.

Baeza-Yates. R. y Ribeiro-Neto, B (1999). *Modern Information Retrieval. Addison-Wesley*. New York. P. 93 – 112.

Davies J., Fensel D., Harmelen F. (2003), *"Toward the Semantic Web. Ontology-driven Knowledge Management"*. Ed. John Wiley&Sons, Ltd.

Antoniou G. and Van Harmelet F. (2004). *A Semantic Web Premier*, the MIT Press, Cambritge Massachusetts, p 1-12Antoniou & Harmelen.

Gruber T. R. 1993. *"Toward principles for the design of ontologies used for knowledge sharing". En: Formal Ontology in Conceptual Analysis and Knowledge Representation.* The Netherlands: Kluwer Academic Publishers.

A. Maedche, B. Motik, and L. Stojanovic. (2003), *Managing multiple and distributed ontologies on the semantic web*, The VLDB Journal 12, no. 4, 286-302.

Gómez-Pérez A., M. Fernández-López, A. de Vicente (1996), *Towards a Method to Conceptualize Domain.*

Fluit, C.; Sabou, M.; van Harmelen, F. (2003). *Supporting User Tasks through Visualization of Light-weight Ontologies. Handbook on Ontologies in Information Systems.* Springer-Verlag.

Contreras, J. (2003). *"Duontology: an Approach to Semantic Portals based on a Domain and Visualization Ontology"*. KTWeb.

Hendler, James, (2001). *"Agents and the Semantic Web"*. En: IEEE Intelligent Systems, marzo/abril, pp. 30-37.

RDF. (2004). *"RDF Semantics". W3C Recommendation.* http://www.w3.org/TR/rdf-mt/. Consultado (Junio 2011).

OWL. (2004). *"Web Ontology Language (OWL)". W3C Recommendation* http://www.w3.org/2004/OWL/. (Marzo 2011).

XML. (2001). *Extensible Markup Language*. http://www.xmlnews.org. (Junio de 2011).

JENA. (2011). *Semantic Web Framework.* http://jena.sourceforge.net/; (Febrero 2011).

MySQL. *http://www.mysql.com/. (Febrero de 2011).*

JAVA. *Java Platform, Standard Edition* http://www.oracle.com/technetwork/java/javase/downloads/index.html (Febrero de 2011).

JADE. *Java Agent Development Framework.* http://jade.tilab.com/ /. (Febrero de 2011).